PR #26884 完整报告

sgl-project/sglang

[AMD] Fix GPT-OSS MXFP4 accuracy on ROCm AITER path

合并时间：2026-06-02 13:30

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/26884

执行摘要

- 一句话：修复 AMD ROCm AITER 上 GPT-OSS MXFP4 精度错误
- 推荐动作：本 PR 展示了系统性的精度调试过程，涉及跨栈的权重布局、kernel dispatch 和属性传播问题。建议团队内对涉及 AITER MXFP4 的后续开发仔细审查 is_shuffled 标志的传播和 GateMode 配置。值得精读，尤其是在理解量化内核集成和平台适配方面。

功能与动机

根据 PR 描述，在 ROCm AITER 上，GPT-OSS MXFP4 的 fused-MoE 路径一直在静默输出错误结果。有两个根本原因：

1) 当前的 shuffle_weight_a16w4 产生了 gate/up 交织的 tile 布局，而 AITER 的 MXFP4 fused-MoE kernel（包括 FlyDSL 和 CK）期望的是分隔布局； 2) 权重打乱后 is_shuffled 标志未正确传递（.data 赋值丢弃了 Python 自定义属性），导致 AITER 分配器选择了错误的 preshuffle_off codegen 分支。这两个 bug 相互作用，最终导致 GSM8K 评分从 ~0.88 跌至 ~0.21。

实现拆解

实现过程包含以下步骤：

1. 修复权重布局与标志（python/sglang/srt/layers/quantization/mxfp4.py）：在 Mxfp4MoEMethod.process_weights_after_loading 和 apply 方法中，先对 w13 权重进行 gate/up 解交织（de-interleave）以匹配 AITER kernel 期望的分隔布局，然后用 shuffle_weight(is_guinterleave=False, gate_up=...) 和 shuffle_scale(is_guinterleave=False, gate_up=...) 替代旧的 shuffle_weight_a16w4 和 shuffle_scale_a16w4。同时显式设置 layer.w13_weight.is_shuffled = True 和 layer.w2_weight.is_shuffled = True，并确保 view 后的新张量重新标记 is_shuffled。
2. 新增环境变量控制布局（python/sglang/srt/environ.py）：在 Envs 类中添加 SGLANG_USE_AITER_MOE_GU_ITLV = EnvBool(True)，默认启用交织布局以保持向后兼容。
3. 调整 AITER run 方法（python/sglang/srt/layers/moe/moe_runner/aiter.py）：在 AiterRunnerCore.run 中，根据该环境变量选择 GateMode.SEPARATED 或 INTERLEAVE，并传入 swiglu_limit。

4. 自动为 GPT-OSS 选择分隔布局（python/sglang/srt/server_args.py）：在 _handle_model_specific_adjustments 中，当检测到 ROCm、AITER 和 MXFP4 量化时，自动将 SGLANG_USE_AITER_MOE_GU_ITLV 设为 False，确保 GPT-OSS 路径始终使用分隔布局。
5. 更新测试（test/registered/amd/accuracy/mi35x/test_gpt_oss_eval_mi35x.py）：在模型配置中显式设置 SGLANG_USE_AITER_MOE_GU_ITLV=0，使测试与实际部署行为一致。

这些修改涉及核心量化逻辑、运行时 kernel 分发、环境变量配置和服务端自动调优，共同解决了两个根本原因。

关键文件：

- python/sglang/srt/layers/quantization/mxfp4.py（模块 量化层；类别 source；类型 core-logic；符号 Mxfp4MoEMethod.process_weights_after_loading, Mxfp4MoEMethod.apply）：核心量化量化方法，修复权重布局和标志设置，是修复的主要位置。
- python/sglang/srt/layers/moe/moe_runner/aiter.py（模块 MOE 执行器；类别 source；类型 dependency-wiring；符号 AiterRunnerCore.run）：AITER MoE 运行时核心，根据环境变量路由 gate_mode。
- python/sglang/srt/environ.py（模块 环境变量；类别 source；类型 core-logic；符号 SGLANG_USE_AITER_MOE_GU_ITLV）：声明新的环境变量，控制 gate/up tile 布局默认值。
- python/sglang/srt/server_args.py（模块 服务配置；类别 source；类型 core-logic；符号 _handle_model_specific_adjustments）：自动为 GPT-OSS 模型设置环境变量为 False，确保使用分隔布局。
- test/registered/amd/accuracy/mi35x/test_gpt_oss_eval_mi35x.py（模块 端到端测试；类别 test；类型 test-coverage；符号 post_init）：更新测试配置，显式设置环境变量以覆盖自动化默认值，验证修复。

关键符号：Mxfp4MoEMethod.process_weights_after_loading, Mxfp4MoEMethod.apply, AiterRunnerCore.run, _handle_model_specific_adjustments, post_init

关键源码片段

python/sglang/srt/layers/quantization/mxfp4.py

核心量化量化方法，修复权重布局和标志设置，是修复的主要位置。

```
if _use_aiter:
    # Bias must be fp32 for the AITER kernels.
    if layer.w13_weight_bias is not None:
        layer.w13_weight_bias.data = layer.w13_weight_bias.data.to(torch.float32)
    if layer.w2_weight_bias is not None:
        layer.w2_weight_bias.data = layer.w2_weight_bias.data.to(torch.float32)

    # HF GPT-OSS 将 w13 存储为 gate/up 交织的行对 [(g0, u0), (g1, u1), ...]。
    # AITER MXFP4 fused MoE 内核（FlyDSL `gate_mode="separated"` 和 CK `preshuffle_on`）
    # 期望分隔布局 [gate_0..gate_{N-1}, up_0..up_{N-1}]。
```

```
# 在 tile shuffle 之前解交织 weights、scales 和 bias，
# 使 shuffle 后的字节落在内核读取的正确布局中。
e, n, k = layer.w13_weight.shape
layer.w13_weight.view(torch.uint8).copy_(
    layer.w13_weight.data.view(torch.uint8)
    .view(e, n // 2, 2, k)
    .permute(0, 2, 1, 3)
    .contiguous()
    .view(e, n, k)
)
layer.w13_weight_scale.data = (
    layer.w13_weight_scale.data.view(e, n // 2, 2, -1)
    .permute(0, 2, 1, 3)
    .contiguous()
    .view(e, n, -1)
)
layer.w13_weight_bias.data = (
    layer.w13_weight_bias.data.view(-1, n // 2, 2)
    .permute(0, 2, 1)
    .contiguous()
    .view(-1, n)
)

# 使用 `is_guinterleave=False` 的 shuffle 以匹配 ATOM 和 tuned FlyDSL 配置。
# 旧的 `shuffle_weight_a16w4` 使用了 gate/up 交织的 tile 布局，导致精度丢失。
layer.w13_weight.data = shuffle_weight(
    layer.w13_weight, is_guinterleave=False, gate_up=True
)
shuffled_w13_scale = shuffle_scale(
    layer.w13_weight_scale.view(-1, layer.w13_weight_scale.shape[-1]),
    experts_cnt=self.num_experts,
    is_guinterleave=False,
    gate_up=True,
)
layer.w2_weight.data = shuffle_weight(
    layer.w2_weight, is_guinterleave=False, gate_up=False
)
shuffled_w2_scale = shuffle_scale(
    layer.w2_weight_scale.view(-1, layer.w2_weight_scale.shape[-1]),
    experts_cnt=self.num_experts,
    is_guinterleave=False,
    gate_up=False,
)

# 必须显式设置 is_shuffled 属性，否则 AITER fused_moe
# 会通过 getattr(w1, "is_shuffled", False) 判断为 False，
# 从而选择错误的 preshuffle_off codegen 分支。
layer.w13_weight.is_shuffled = True
layer.w2_weight.is_shuffled = True
```

```
        layer.w13_weight_scale = torch.nn.Parameter(
            shuffled_w13_scale, requires_grad=False
        )
        layer.w2_weight_scale = torch.nn.Parameter(
            shuffled_w2_scale, requires_grad=False
        )
```

python/sglang/srt/layers/moe/moe_runner/aiter.py

AITER MoE 运行时核心，根据环境变量路由 gate_mode。

```
    if quant_info.swiglu_limit > 0:
        # 默认使用 INTERLEAVE 以保持向后兼容，但可通过环境变量切换为 SEPARATED。
        # Mxfp4MoEMethod (gpt-oss MXFP4) 和 tuned FlyDSL 内核使用 SEPARATED 布局。
        extra["gate_mode"] = (
            GateMode.INTERLEAVE.value
            if envs.SGLANG_USE_AITER_MOE_GU_ITLV.get()
            else GateMode.SEPARATED.value
        )
        extra["swiglu_limit"] = quant_info.swiglu_limit
```

python/sglang/srt/environ.py

声明新的环境变量，控制 gate/up tile 布局默认值。

```
    # AMD & ROCm
    SGLANG_USE_AITER = EnvBool(False)
    SGLANG_USE_AITER_UNIFIED_ATTN = EnvBool(False)
    # 选择 AITER MoE gate/up tile 布局：True -> 交织（匹配 FlyDSL `gate_mode="interleave"`），
    # False -> 分隔（匹配 `gate_mode="separated"`，用于 gptoss_fp4 tuned 配置和
    Mxfp4MoEMethod 的权重 shuffle）。
    SGLANG_USE_AITER_MOE_GU_ITLV = EnvBool(True)
    SGLANG_ROCM_FUSED_DECODE_MLA = EnvBool(False)
```

评论区精华

在 review 过程中，审核者 HaiShaw 提出关键要求：“Let try to set SGLANG_USE_AITER_MOE_GU_ITLV to false in gptoss + hip + aiter path?” 此要求被采纳，通过后续提交在 server_args.py 中自动设置该环境变量为 False，确保 GPT-OSS 路径默认使用分隔布局。审核者在后续迭代后批准了该 PR。

- 设置 SGLANG_USE_AITER_MOE_GU_ITLV 默认值 (design): 已采纳，在 server_args.py 中自动设为 False。

风险与影响

- 风险：主要风险包括： 1) 布局不兼容：环境变量 SGLANG_USE_AITER_MOE_GU_ITLV 默认 True（交织布局），但 GPT-OSS 路径通过 server_args.py 自动设为 False。如果其他模型也使用 AITER MXFP4 且需要交织布局，现有关键路径仍保持默认，但任何新增的 MXFP4 模型必须确保 shuffle 阶段与 gate_mode 对齐。 2) 属性传播遗漏：虽然本 PR 修复了 .data 赋值丢失 is_shuffled 属性的问题，但其他类似模式（如 view 后未重新标记）

可能存在于代码其他部分。 3) 测试覆盖：当前测试仅覆盖 GPT-OSS 模型和特定的 FlyDSL/CK kernel 组合，未覆盖其他 AITER MXFP4 路径（如其他模型的交织布局）。 4) 环境变量配置项增加：新增的环境变量需要团队理解其含义，否则可能误用。

- 影响：用户影响：修复了 ROCm AITER 上 GPT-OSS MXFP4 路径的精度问题，GSM8K 指标恢复到与 ATOM 基线一致（0.88）。对不涉及 ROCm/AITER 的用户无影响。系统影响：无性能回退，仅改变权重布局和 kernel 选择；新增环境变量默认保留原有行为。团队影响：维护者需注意新增的环境变量及其在不同模型组中的默认值设定；社区贡献者引入新的 AITER MXFP4 模型时必须确认布局一致性。总体影响限于特定硬件和模型组合，但修复了关键精度问题。
- 风险标记：核心路径变更，环境变量配置依赖，平台特定修复，布局对齐风险

关联脉络

- 暂无明显关联 PR